**I termin**

**1. Napisz czy kod jest poprawny. Ile razy uruchomi się konstruktor T?**

int main()

{

set<T, T::op> s(10); // 1

vector<T> v; // 2

v.reserve(10); // 3

T t[10]; // 4 10 konstruktorów tylko tu

}

**2. Napisz czy kod jest poprawny:**

int main()

{

struct A { virtual void a(); static void s(); }; // 1

struct B : public A { virtual void b(); }; // 2

const A a; A & b = B(); // 3

a.a(); // 4

a.s(); // 5

a.b(); // 6 X

b.a(); // 7

b.s(); // 8

b.b(); // 9

}

**3. Uzupełnij**

/\* Uzupełnij \*/

int main()

{

array<box<>, 2 > a;

double i = a[0] = a[1] = 1; // a == {1,1}

const array<box<int>> b = a;

b[0] = b[1] = 2; // b == {1,1}

}

**4. Uzupełnij**

/\* uzupełnij 1\*/

int main()

{

deque<int> c;

generate\_n(/\* uzupełnij 2\*/, 10, gen::next), gen(); // uzupełnia 1,2,...,10

c./\* uzupełnij 3\*/; // usuwa parzyste, zostaje 1, 3, 5, 7, 9

// w /\* uzupełnij 3 \*/ używamy TYLKO obiektów funkcyjnych z STL!

}

**5. Uzupełnij**

/\* Uzupełnij \*/

struct A { };

int main()

{

STATIC\_ASSERT(is\_ptr<A\*>::value) // OK

STATIC\_ASSERT(is\_ptr<remove\_ptr<A\*>::value\_type\*>::value) //OK

STATIC\_ASSERT(is\_ptr<remove\_ptr<A\*>::value\_type>::value) //Błąd, bo value\_type to nie wskaźnik tylko obiekt (?)

}

**II termin**

**1. Napisz czy kod jest poprawny. Ile razy i jaki uruchomi się konstruktor T?**

int main()

{

set<T, T::op> s(10); // 1

vector<T> v(); // 2

v.resize(10); // 3

T \* t[10]; // 4

}

**2. Napisz program, w którym trzeba skorzystać z dynamic\_cast**

**3. Uzupełnij**

/\* uzupełnij \*/

box<> a = 2.2;

box<int> b = 3;

box<char> c = box<char>(‘A’);

//box<char> d = ‘A’;

//odkomentowanie tej linijki powoduje blad kompilacji

int main()

{

a.print(); // Wypisuje obiekt T: 2.2

b.print(); // Wypisuje obiekt T: 3

c.print(); // Wypisuje obiekt char: 65

}

**4. Uzupełnij**

/\* uzupelnj 1 \*/

int main()

{

list<el\_type> v;

std::ostream o = std::cout

fill\_n(/\* uzupelnij 2 \*/, 10, el\_type()); // uzupelnia zerami

o << print\_stl\_container(v.begin, v.end);

// wynik programu: 0,0,0,0,0,0,0,0,0,0

}

**5. Uzupełnij tak żeby program wyliczał wielkość tablicy podczas kompilacji**

/\* uzupełnij \*/

int main()

{

int tab[product\_1\_n<sum<1,2,3>::n>::n];

}